Introdução de Algoritmos e Raciocínio Lógico

Para criar programas computacionais é necessário empregar a lógica de programação, uma técnica onde se faz necessário o desenvolvimento de competências para interpretação, abstração, resolução e formalização de problemas. Trata-se da base da ciência da computação onde sequências lógicas de passos são aplicados a fim de fornecer uma solução a um determinado problema ou para executar uma tarefa específica.

Para tal, é aplicado o desenvolvimento de algoritmos que são passo a passos ou sequências lógicas de instruções bem definidas para atingir um objetivo claro, não ambíguo e em um tempo finito. Os algoritmos podem ser expressos por meio de linguagem natural, fluxogramas e pseudocódigo, por exemplo. Eles são os primeiros passos de um programa (software) que são traduzidos em uma linguagem de programação qualquer, como Python, C, Java, PHP e Java Script, para então ser entendidos pelo computador na execução do programa ou na exibição de uma página Web. Um exemplo clássico e simplificado da criação de um algoritmo são as receitas culinárias, como os passos a passos realizados por uma pessoa na concepção de um bolo.

Nesta unidade de aprendizagem, estudaremos os conceitos básicos que envolvem algoritmos e lógica de programação, o funcionamento básico de um computador e as etapas básicas para o desenvolvimento de um algoritmo.

Bons estudos.

Ao final desta Unidade de Aprendizagem, você deve apresentar os seguintes aprendizados:

- Definir o conceito de algoritmos e lógica de programação.
- Diferenciar os componentes básicos de funcionamento de um computador na execução de algoritmos.
- Aplicar as etapas de construção de um algoritmo na solução de problemas.

INFOGRÁFICO

O esquema a seguir mostra o conteúdo que será abordado nesta unidade de aprendizagem.

Conteúdo interativo disponível na plataforma de ensino!

CONTEÚDO DO LIVRO

Cada vez mais, o ser humano cria máquinas que o auxiliem em seu trabalho para, assim, diminuir os seus esforços e economizar tempo em suas atividades.

Dentre essas máquinas, o computador vem se mostrando uma das mais versáteis, rápidas e seguras. No entanto, para que ele trabalhe, é necessário que receba instruções através de programas, sendo possível, desta forma, receber, manipular e armazenar dados.

As instruções devem ser projetadas e desenvolvidas por programadores, que desenvolvem várias aplicações, como editores de texto, planilhas de cálculo, navegadores para internet, jogos, etc.

Para auxiliá-lo, acompanhe o capítulo Introdução à lógica de programação da obra *Algorítmos de programação*. O livro servirá como base para a nossa unidade de aprendizagem.

ALGORÍTMOS DE PROGRAMAÇÃO

Marcela Santos

Introdução à lógica de programação

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Conhecer o conceito de algoritmo: apresentação, exemplos e definição.
- Entender os conceitos de comandos, variáveis, expressões e instruções.
- Compreender as formas de execução de um programa: compilação e interpretação.

Introdução

Já parou para pensar como é feito um jogo, um *site* da Internet ou um aplicativo do seu dispositivo móvel? Cada vez mais, usamos tecnologia desde o nosso trabalho até a nossa diversão. E que tal você aprender como os programas são desenvolvidos para poder criar seus próprios jogos, *sites*, etc.? A base fundamental de um programa de computador são os algoritmos e a lógica de programação. Esses conceitos, além de serem bases da programação de computadores, desenvolvem habilidades, como raciocínio lógico, que podem ser utilizadas na resolução dos mais diversos problemas nas mais diversas áreas.

Neste capítulo, você vai acompanhar a construção de um algoritmo, os conceitos básicos envolvidos e como funciona a execução de um programa de computador.

Algoritmo: apresentação, exemplos e definição

Um algoritmo nada mais é do que um conjunto de passos para a realização de uma determinada tarefa. O que é importante entendermos é que um algoritmo não precisa ser necessariamente algo ligado à tecnologia. Na realidade, o tempo todo estamos criando algoritmos para realizarem diversas tarefas. Quer ver um exemplo?

Quando você vai sair de casa a caminho da faculdade, existe uma sequência de passos que você realiza, que tem uma ordem. As ordens das coisas que você precisa fazer para chegar à faculdade são escolhidas para te ajudar a realizar a tarefa: "ir para a faculdade".

Outros exemplos:

- Preparar um hambúrguer.
- Fazer um avião de papel.
- Trocar o pneu de um carro.

Exemplo

Exemplo — Preparar um hambúrguer

1. Cortar o pão e deixá-lo aberto com as partes de dentro voltadas para cima;
2. lavar a alface e o tomate;
3. fritar o hambúrguer;
4. colocar o hambúrguer sobre uma das partes do pão que está aberta;
5. colocar uma fatia de queijo, uma rodela de tomate e uma folha de alface sobre o hambúrguer;
6. fechar o pão com a outra parte que ficou sem nada sobre ela.

Pronto — sanduíche prontinho! Mas podemos fazer algumas perguntas sobre essa nossa aventura na cozinha:

- Existe somente um jeito de fazer esse hambúrguer?
- A ordem das atividades é importante?

Não existe uma única forma; o importante é que essas atividades sejam organizadas de tal forma que a tarefa "preparar um hambúrguer" seja realizada. Nesse momento, o importante é fazer, não importando se mais rápido, mais gostoso ou com mais recheio. Assim, a ordem é de extrema importância para que consigamos chegar ao objetivo final.

Afinal, o que isso tem a ver com computador? Bom, primeiro é preciso entender por que o uso de computador para resolução dos mais diversos problemas e das mais diversas tarefas tornou-se tão imprescindível. Para isso, vamos comparar o ser humano a um computador.

O humano é um ser inteligente, capaz de realizar diversas tarefas, como, por exemplo, calcular o valor do desconto de um jogo que está prestes a comprar. Já um computador é uma máquina "burrinha", que só executa o que mandamos. Em contrapartida, ele não se cansa. Imagina se você tivesse que calcular, durante o dia inteiro, o desconto da venda de uma loja? Poderia cansar-se e até cometer um erro, não é mesmo?

Então, é por isso que, desde quando foi criado até hoje, o papel fundamental do computador é ajudar o ser humano nas suas tarefas. Mas quem ensina o computador o que ele tem que fazer? Esse papel é do ser humano, que o faz por meio dos programas.

Os programas de computador são conjuntos de instruções que o computador precisa realizar para poder concluir uma determinada tarefa. Então, um algoritmo, em computação, também é um conjunto de passos, só que o foco é desenvolver um programa de computador.

Um algoritmo bastante famoso é o utilizado pelo Google, para realizar suas buscas. Ele foi desenvolvido a partir de 1995, pelos então estudantes Larry Page e Sergey Brin. Imagine que você tenha que realizar uma busca por nome em uma agenda telefônica — fica mais fácil se a agenda estiver organizada por nome, concorda? Então, os primeiros sistemas de busca ordenavam os *sites* dessa forma, sendo que o grande diferencial do Google foi utilizar um algoritmo que ordenasse os *sites* do mais acessado para os menos acessados. Dessa forma, a probabilidade do que o usuário estava procurando estar entre os primeiros é maior.

Os algoritmos estão em várias áreas — medicina, bolsa de valores, mobilidade urbana. Seu uso está cada vez mais difundido, e aprender como escrever um algoritmo bem como torná-lo mais eficiente é uma habilidade muito importante para os profissionais das mais diversas áreas.

Fique atento

Algoritmo pode ser definido como uma sequência de passos que visam a atingir um objetivo definido (FORBELLONE; EBERSPÄCHER, 2005).

Comandos, variáveis, expressões e instruções

No nosso exemplo do hambúrguer, falta pensarmos em uma coisa: saber como escrever e definir quais são os passos necessários. Bom, cada pessoa pode definir de um jeito, e isso poderia acontecer também ao escrevermos um algoritmo para um programa de computador, o que tornaria o trabalho dos computadores muito mais complicado, pois cada pessoa poderia definir esses passos de uma forma. Em computação, costuma-se padronizar grande parte dos conceitos envolvidos, e a definição dos passos de um algoritmo também foi padronizada.

Por meio de comandos, podemos formar as instruções e as expressões. Assim, os comandos são os tijolos da nossa construção; com eles, o computador entende o que deve fazer. Escrever um texto na tela, por exemplo, pode ser a ação que será executada pelo comando "escrever <texto" que será escrito na tela.

Existem vários tipos de comandos: comandos para atribuição de valor, comandos para entrada e saída de dados, comandos para estruturas de seleção ou de repetição — vamos conhecer todos eles. Agora, você precisa saber que comando denota uma ação (ou um conjunto delas) que o computador irá executar, tem forma e segue algumas regrinhas.

Outra definição que usaremos ao longo de todo o nosso curso é a de variáveis. Durante a execução de um programa, dados são entregues ao computador, que, executando esse programa, modifica os dados e "responde" conforme o esperado.

Exemplo

Exemplo — Operação de saque em um autoatendimento

1. Entrar com os dados de conta e senha;
2. solicitar qual opção o cliente deseja realizar;
3. se for saque, solicitar o valor a ser sacado;
4. se existir saldo, entregar o dinheiro;
5. se não, mostrar a mensagem de "saldo indisponível" na tela;
6. encerrar a operação.

Esse valor que você solicitou é um dado que precisa ser armazenado na memória do computador. Essa região da memória do computador, onde guardamos, mesmo que temporariamente, o dado, chamamos de "variável", e, como o próprio nome diz, pode sofrer mudança (variação) de valor.

Saiba mais

Memória de um computador é o meio físico para armazenar dados temporariamente ou permanentemente. Para saber mais, consulte TANENBAUM, 2000.

Para podermos usar as variáveis, precisamos inicialmente criá-las. Em programação, a criação de uma variável é chamada de "declaração de uma variável", que pode ser comparada ao ato de etiquetar pequenas gavetas para organizar diversos itens. A grande ideia por detrás da variável é que cada uma delas tem um nome e um tipo. Assim, somente os dados daquele tipo podem ser colocados na sua respectiva variável. Não daria para colocar um grampeador em uma gaveta etiquetada com canetas. Veremos mais detalhes sobre variável adiante.

Com as variáveis e os comandos, podemos formar as instruções e/ou expressões. Uma expressão é uma combinação de variáveis, comandos e operadores. Já uma instrução é uma operação única, que é executada pelo processador do computador. Assim, o programa de computador é formado por várias instruções. Em resumo, as instruções são formadas por comandos e/ou variáveis. Todas essas ferramentas são usadas para escrever um programa, utilizando o conceito de algoritmo.

Representação de um algoritmo

Como foi visto, um algoritmo é um conjunto de passos para que uma tarefa seja realizada. Esses passos são organizados de forma lógica, a fim de se chegar ao objetivo final. Existem várias formas de se representar um algoritmo — até agora usamos o português para isso, mas existem representações gráficas: fluxograma e diagrama de Chapin. Vamos representar o algoritmo do saque, usando o fluxograma da Figura 1:

INÍCIO

Solicitar dados do usuário

Solicitar a opção de operação

Se for saque

V

Solicitar valor a ser sacado

Saldo suficiente?

F

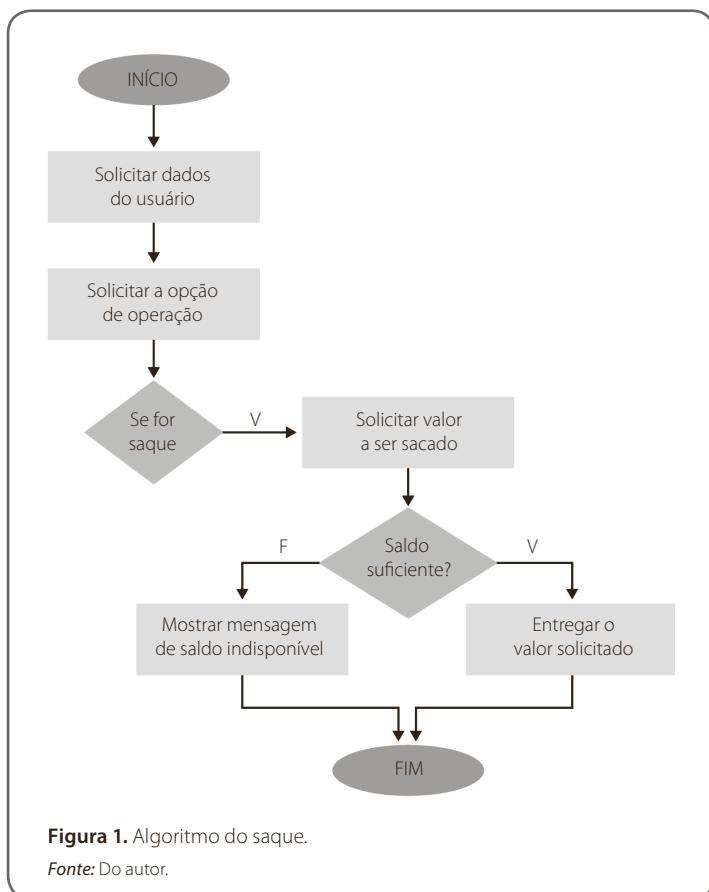
V

Mostrar mensagem de saldo indisponível

Entregar o valor solicitado

FIM

Figura 1. Algoritmo do saque.

Fonte: Do autor.

As representações gráficas são mais fáceis de entender devido à sua própria natureza. A lógica envolvida fica mais clara, porém é preciso que se aprenda a padronização dos itens que podem ser usados. Por esse motivo, ao logo deste livro, usaremos o português para a representação dos algoritmos.

Saiba mais

Para saber mais sobre a representação de algoritmos, consulte FORBELLONE; EBERSPÄCHER, 2005.

Execução de um programa: compilação e interpretação

Bom, já sabemos que um programa é um conjunto de instruções que são entregues ao computador para que ele realize determinada tarefa. Essa ação de realizar uma tarefa em computação chamamos de "execução de um programa", e existem duas maneiras de um programa ser executado. Antes de passarmos para essa etapa, é importante dizer que os algoritmos, mesmo que padronizados, ainda não estão escritos em uma linguagem que o computador entenda. Existem várias linguagens que o computador entende: C, Java, Python são exemplos de linguagens de programação. Então, o processo de desenvolver um programa passa por estas etapas:

1. Entender o problema ou a tarefa que queremos que o programa execute.
2. Desenvolver um algoritmo.
3. Escrever o algoritmo em uma linguagem de programação — esse arquivo com o algoritmo escrito em uma linguagem de programação é chamado de "código-fonte".
4. Enviar o programa para que o processador o execute, compilando ou interpretando o código-fonte.
   - Compilar um arquivo é traduzir de uma linguagem para outra. Assim, quando compilamos um código-fonte, estamos transformando um arquivo escrito em uma linguagem de programação em uma linguagem que o processador pode executar, também chamada de linguagem de máquina. Exemplos de linguagens compiladas: C, C++, Erlang, Haskell.
   - Outras linguagens de programação são interpretadas, funcionando da seguinte forma: o código fonte é passado para o interpretador, que pega cada instrução escrita no código-fonte e a executa de forma direta, não realizando uma tradução de todo o código-fonte. Exemplos de linguagens interpretadas: Java, Python, Ruby, PHP.
   - A maior vantagem de uma linguagem compilada é a velocidade de execução. Como, nesse tipo de linguagem, o código-fonte é convertido diretamente em uma linguagem de máquina, a execução é mais rápida e eficiente quando comparada com uma linguagem interpretada especialmente em sistemas onde a complexidade é alta.
   - Já as linguagens interpretadas não precisam ser totalmente traduzidas para um código de máquina, pois cada uma das instruções é executada diretamente pelo seu interpretador. Logo quando foram

inventadas essas linguagens, a velocidade do tempo de execução, quando comparadas com as linguagens compiladas, era, sem sombra de dúvidas, muito alta e a principal desvantagem quando feita essa comparação. Porém, com a invenção da tecnologia *just-in-time* (JIT), essa desvantagem continua existindo, mas bem menor. Essa técnica consiste em realizar uma tradução dinâmica, em tempo de execução e não antes da execução. Assim, a diferença de execução entre linguagens compiladas e interpretadas diminuiu.

- Em contrapartida, um programa compilado precisa ser gerado para cada plataforma que for rodar: Windows, Linux, Mac OS. Já um programa interpretado pode rodar em qualquer máquina, já que sua "interpretação" é feita em tempo de execução, sendo que a única necessidade é ter o interpretador na máquina onde esse programa será executado.

Link

Para saber mais sobre JIT, consulte:

https://goo.gl/GjiWhs

Referências

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. *Lógica de programação*: a construção de algoritmos e estrutura de dados. São Paulo: Makron Books, 2005.

TANENBAUM, A. S. *Sistemas operacionais*. Porto Alegre: Bookman, 2000.

DICA DO PROFESSOR

Conhecer um pouco sobre o funcionamento do computador e sobre as etapas de desenvolvimento de um algoritmo é muito importante a fim de que possamos desenvolver as habilidades para a resolução de problemas através de algoritmos com eficiência e eficácia. Assista ao vídeo para conhecer um pouco sobre este tema.

Conteúdo interativo disponível na plataforma de ensino!

EXERCÍCIOS

1) Analise as características apresentadas pelos algoritmos em cada alternativa e selecione a alternativa que não representa um algoritmo, ou seja, não atende aos requisitos para ser considerado um algoritmo.

A) Calcular todos os números primos de 2 até 100.

B) Encontrar e escrever todos os termos da sequência de Fibonacci.

C) Calcular a soma de todos os valores no intervalo de 100 até 1000 que são divisíveis por 3 .

D) Calcular a média de todos os valores pares entre 100 e 200, inclusive.

E) Encontrar e escrever os 10 primeiros termos da sequência de Fibonacci.

2) A lógica de programação é muito importante para o desenvolvimento de algoritmos, que são a descrição de uma sequência finita de instruções ou operações que devem ser executadas em uma determinada ordem e em tempo finito para assim atingir o resultado esperado, ou seja, a resolução de um problema. Considerando o conceito de algoritmos apresentado acima, avalie as afirmações a seguir.

I) A sequência lógica é a execução das operações/instruções na ordem determinada para atingir um objetivo ou chegar a uma solução de um problema.
II) Um algoritmo possui somente uma única sequência lógica de instruções para a sua solução.
III) A lógica de programação é a técnica de encadear pensamentos para atingir determinado objetivo.
IV) A instrução possui um conjunto de regras/normas definidas para a realização de alguma atividade que resulta em uma ação.

É correto apenas o que se afirma em:

A) I e II.

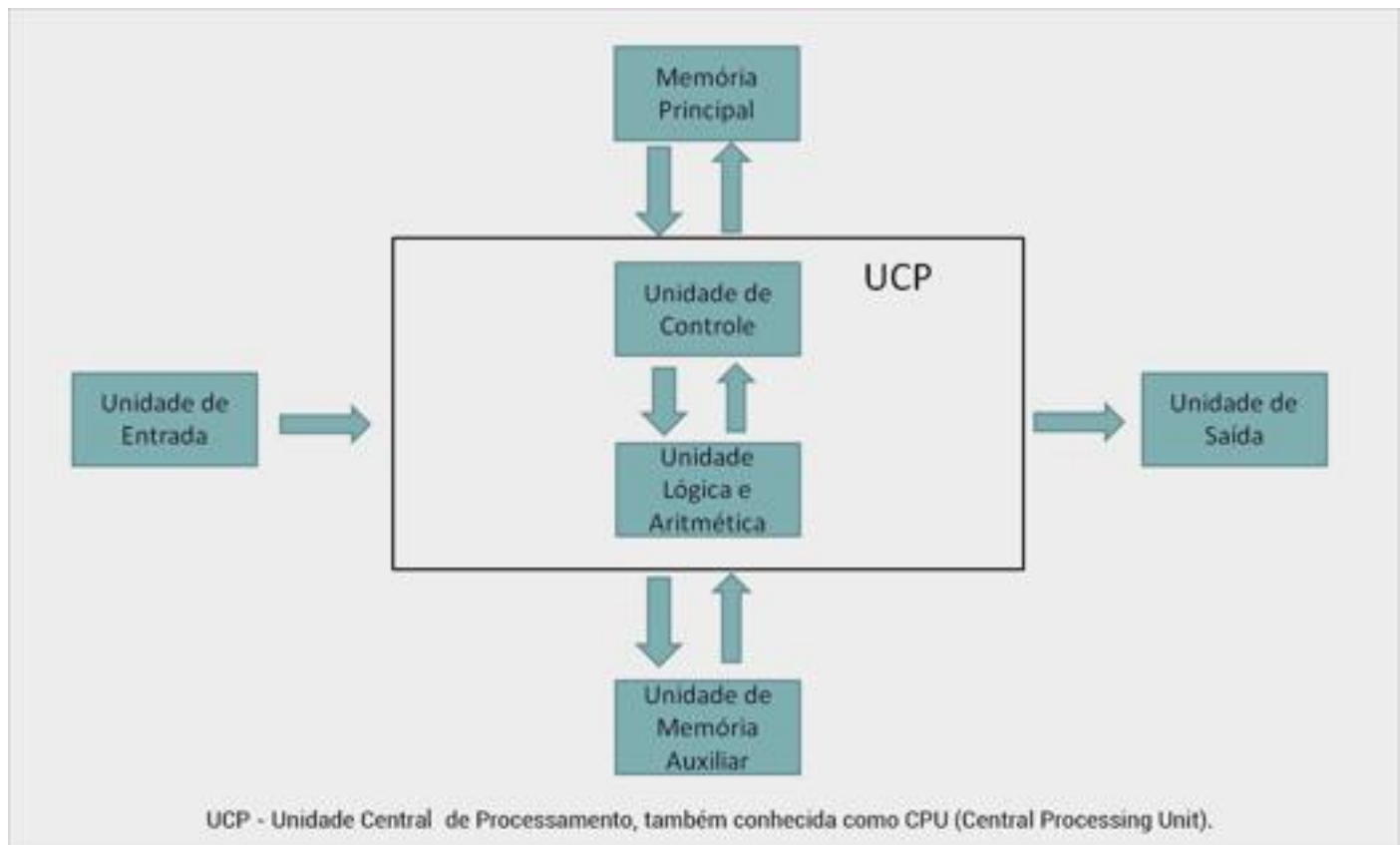
B) I e III.

C) II e IV.

D) I, III e IV.

E) II, III e IV

3) **O computador é um dispositivo capaz de realizar cálculos e de tomar decisões lógicas em velocidades muito mais rápidas que os seres humanos. Além disso, processam dados através de conjuntos de instruções denominados de programas, chamados de softwares. Não importa a aparência física, um computador é composto por vários dispositivos, como teclado, monitor, mouse, disco rígido, memória, unidade de processamento, os quais são denominados hardware. A figura a seguir representa a estrutura básica de um computador. Analise a figura e selecione o componente que representa a parte de "processamento" do computador, responsável por executar cálculos como adição, subtração, multiplicação e divisão e mecanismos de tomada de decisão, que permitem ao computador, por exemplo, comparar dois itens na unidade de memória e determinar se eles são ou não iguais. O outro componente é responsável pelo tráfego de dados, obtendo dados da memória e os interpretando, e controlando a transferência de dados da memória para a**

ULA, da entrada para a memória e da memória para a saída.

Qual o componente que representa a funcionalidade descrita acima?

A) Unidade de entrada.

B) Unidade de saída.

C) As unidades de memória auxiliar.

D) A Unidade Lógica e Aritmética (ULA, ou ALU, Arithmetic and Logic Unit) e a Unidade de Controle (UC).

E) A memória principal.

4) **Muitas pessoas encontram dificuldades em converter valores informados em Reais**

(R$) para valores em Dólar ($). Selecione a alternativa correta que representa os componentes de entrada, processamento e saída do algoritmo que realize a conversão.

A) Entrada: valor em reais e o valor de 1 dólar.
Processamento: multiplicar o valor em reais pelo valor do dólar diário.
Saída: mostrar o valor em dólar encontrado.

B) Entrada: valor em reais e o valor de 1 dólar.
Processamento: dividir o valor em reais pelo valor do dólar diário.
Saída: mostrar o valor em dólar encontrado.

C) Entrada: valor em reais e o valor de 10 dólares.
Processamento: subtrair o valor em reais pelo valor do dólar diário.
Saída: mostrar o valor em dólar encontrado.

D) Entrada: valor em reais e o valor de 1 dólar.
Processamento: somar o valor em reais pelo valor do dólar diário.
Saída: mostrar o valor em dólar encontrado.

E) Entrada: valor em reais e o valor de 100 dólares.
Processamento: dividir o valor em reais pelo valor do dólar diário.
Saída: mostrar o valor em dólar encontrado.

5) **A construção de um algoritmo para representar a situação de uma aplicação do mundo real deve ser feita com muito cuidado a fim de que realmente execute as tarefas que se quer de forma correta e em tempo hábil. A dificuldade em gerar bons algoritmos/programas levou à definição de técnicas específicas que iniciam frequentemente com a construção de um algoritmo. Para garantir a qualidade de um programa, deve-se construí-lo seguindo uma série de etapas para, assim, chegar ao produto final: um programa que execute as funcionalidades necessárias à aplicação. Identifique e selecione a alternativa que representa as etapas e a ordem correta de construção de um programa.**

A) Análise do problema -Identificação das entradas e saídas – Validação do algoritmo – Construção do algoritmo – Construção do programa – Teste do programa - Manutenção.

B) Análise do problema -Identificação das entradas e saídas – Construção do algoritmo – Construção do programa – Validação do algoritmo –Teste do programa - Manutenção.

C) Análise do problema -Identificação das entradas e saídas – Construção do algoritmo – Validação do algoritmo – Construção do programa – Teste do programa - Manutenção.

D) Análise do problema -Identificação das entradas e saídas – Construção do algoritmo – Validação do algoritmo - Manutenção do algoritmo – Teste do programa - Construção do programa.

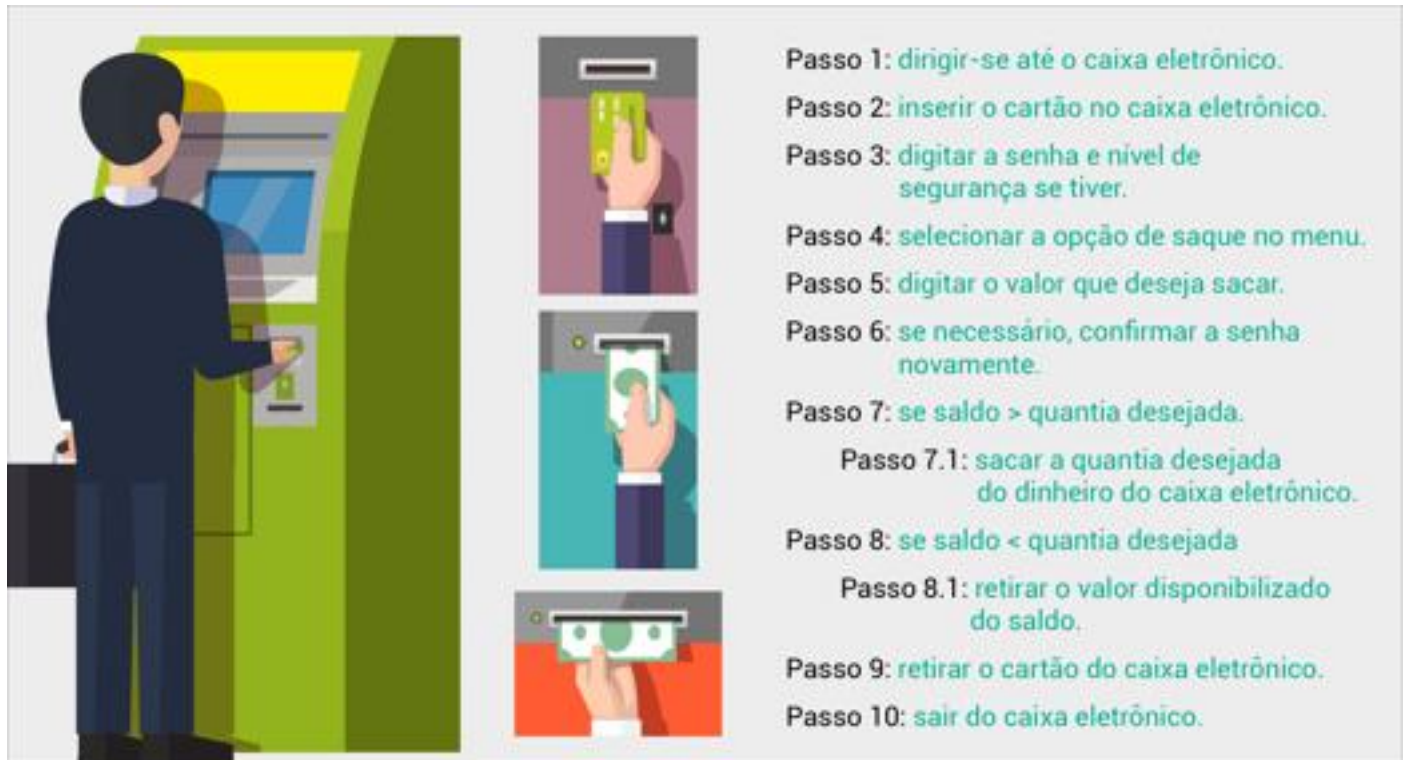
E) Análise do problema - Identificação das entradas e saídas – Construção do programa – Teste do programa - Construção do algoritmo – Validação do algoritmo – Manutenção.

NA PRÁTICA

Você já desenvolveu um algoritmo hoje? Pode ter certeza que sim. Você desenvolveu não somente um, mas vários algoritmos!!!

Vamos ver agora alguns casos práticos da aplicação dos algoritmos no nosso dia a dia.

Todos nós sabemos construir algoritmos. Caso contrário, não conseguiríamos sair de casa pela manhã, ir ao trabalho, realizar compras, decidir qual o melhor caminho para chegar a um lugar, voltar para casa, etc. Para que tudo isso possa ser realizado, é necessário uma série de entradas, como que hora acordar, que hora sair de casa, qual o melhor meio de transporte, qual o melhor trajeto conforme o horário do dia, etc.

Veja a sequência de atividades para que possamos realizar um saque em um caixa eletrônico:

Assim como o saque apresentado acima, construímos e executamos vários algoritmos durante o nosso dia a dia e nem percebemos.

Outro exemplo de aplicação - que, particularmente, não é uma atividade agradável de executar - é o caso de, ao estarmos dirigindo em plena avenida, de repente ..ops....furar o pneu! E, então, como trocar, qual a sequência lógica para que tenhamos sucesso nesta empreitada?

Mesmo não sendo uma atividade muito agradável, vamos lá!

Passo 1: parar o carro.
Passo 2: descer do carro.
Passo 3: abrir o porta-malas e retirar o triângulo e colocar a sinalização.
Passo 4: retirar o macaco, a chave de rodas e o pneu reserva.
Passo 5: afrouxar ligeiramente as porcas.
Passo 6: suspender o carro com o macaco.
Passo 7: retirar as porcas e o pneu.
Passo 8: colocar o pneu reserva.
Passo 9: apertar as porcas.
Passo 10: baixar o carro.
Passo 11: dar o aperto final nas porcas.
Passo 12: recolher e guardar o triângulo, o macaco e o pneu furado.

SAIBA MAIS

Para ampliar o seu conhecimento a respeito desse assunto, veja abaixo as sugestões do professor:

Lógica de Programação - Apresentação e Definições

Neste vídeo introduzimos o curso de Lógica de Programação trazendo alguns conceitos iniciais importantes.

Conteúdo interativo disponível na plataforma de ensino!

Raciocínio Lógico nas Escolas: Uma Introdução ao Ensino de Algoritmos de Programação

O presente artigo relata a experiência de discentes do curso de Licenciatura em Computação, intitulada “Oficina de Raciocínio Lógico e Programação", com objetivo de auxiliar os alunos do 8o e 9o ano na Olimpíada Brasileira de Matemática das Escolas Públicas (OBMEP), e introduzir a Computação na Educação Básica.

Conteúdo interativo disponível na plataforma de ensino!

Complexidade de Algoritmos

A obra apresenta a complexidade de algoritmos, métodos e problemas com ênfase em conceitos e seu emprego em análise e projeto. O livro também analisa as limitações dos problemas em relação à complexidade dos algoritmos que os resolvem e define as principais classes de problemas: P, NP e NP-completa.Os autores, reconhecidos e premiados pela comunidade científica, têm larga experiência de ensino e pesquisa nas melhores universidades do país e também no exterior.